WHAT IS CLAIMED IS:

1. A task-based chip-level hardware architecture, comprising:
a task manager for managing a task with task information; and
a task module operatively connected to the task manager for performing the task in accordance with the task information.

2. The architecture of claim 1, further comprising a plurality of task modules that are in operative communication with the task manager, wherein the architecture is structured according to a star topology.

3. The architecture of claim 2, wherein the task manager is a hub of the star topology.

4. The architecture of claim 1, wherein the task is a predefined function dedicated to the task module.

5. The architecture of claim 1, wherein the task module communicates only with the task manager.

6. The architecture of claim 1, wherein the task module operatively connects to the task manager via a dedicated port.

7. The architecture of claim 1, wherein the task module functions independently of a topology in which it is utilized.

8. The architecture of claim 1, wherein the task manager performs a switching function by switching task management among a plurality of task modules, each of the plurality of task modules performing a predefined task.

9. The architecture of claim 1, wherein the task information is exchanged between the task manager and the task module via a 32-bit bus.

10. The architecture of claim 1, wherein the task information is transmitted from a sender only when a recipient permits transmission therefrom.

11. The architecture of claim 10, wherein when the sender is the task manager, the recipient is the task module, and when the sender is the task module, the recipient is the task manager.

12. The architecture of claim 1, further comprising an auxiliary module that provides a support function to the task module.

13. The architecture of claim 1, further comprising an auxiliary module that provides a support function to the task module, and which auxiliary module communicates only with the task module.

14. The architecture of claim 1, wherein the task information includes a task message having header data and payload data, which header data includes dispatch information for routing the task message, and which payload data includes packet information associated with packet data on which the task is performed.

15. The architecture of claim 14, wherein the dispatch information is made available in the architecture by including at least one of embedding a task list in the header, storing a mapping table in the task manager, and storing the mapping table in the task module.

16. The architecture of claim 1, wherein the task module further comprises a core sublayer for performing the task, which core sublayer includes one or more state machines and accompanying logic to perform the task, and interfaces with one or more auxiliary modules.

17. The architecture of claim 1, wherein the task module further comprises a message processor that provides a messaging interface between the task manager and the task module.

18. The architecture of claim 1, wherein the architecture is structured according to a tree topology such that a plurality of task modules are provided in communication with a plurality of task managers.

19. The architecture of claim 18, wherein the plurality of task managers include a first task manager and a second task manager that are in direct communication with one another to facilitate completion of the task.

20. The architecture of claim 1, further comprising a plurality of task modules operatively communicating over respective ports with the task manager, which task manager receives input data and manages processing of the input data into output data by selectively routing input data information in the form of task messages through one or more of the plurality of task modules to generate the output data.

21. A method of task-based chip-level data processing, comprising:

providing a task manager for managing a task with task information; and

operatively connecting a task module to the task manager for performing the task in accordance with the task information.

22. The method of claim 21, further comprising a plurality of task modules in the step of connecting that are in operative communication with the task manager and structured according to a star topology.

23. The method of claim 22, wherein the task manager in the step of providing is a hub of the star topology.

24. The method of claim 21, wherein the task in the step of providing is a predefined function dedicated to the task module.

25 The method of claim 21, wherein the task module in the step of connecting communicates only with the task manager.

26. The method of claim 21, wherein the task module in the step of connecting operatively connects to the task manager via a dedicated port.

27. The method of claim 21, wherein the task module in the step of connecting functions independently of a topology in which it is utilized.

28. The method of claim 21, wherein the task manager in the step of providing performs a switching function by switching task management among a plurality of task modules, each of the plurality of task modules performing a predefined task.

29. The method of claim 21, wherein the task information in the step of providing is exchanged between the task manager and the task module via a 32-bit bus.

30. The method of claim 21, wherein the task information in the step of providing is transmitted from a sender only when a recipient permits transmission therefrom.

31. The method of claim 30, wherein when the sender is the task manager, the recipient is the task module, and when the sender is the task module, the recipient is the task manager.

32. The method of claim 21, further comprising the step of providing an auxiliary module that provides a support function to the task module.

33. The method of claim 21, further comprising the step of providing an auxiliary module that provides a support function to the task module, and which auxiliary module communicates only with the task module.

34. The method of claim 21, wherein the task information the step of providing includes a task message having header data and payload data, which header data includes dispatch information for routing the task message, and which payload data includes packet information associated with packet data on which the task is performed.

35. The method of claim 34, wherein the dispatch information is made available by further including at least one of the steps of,

embedding a task list in the header,

storing a mapping table in the task manager, and

storing the mapping table in the task module.

36. The method of claim 21, wherein the task module in the step of connecting further comprises a core sublayer for performing the task, which core sublayer includes one or more state machines and accompanying logic to perform the task, and interfaces with one or more auxiliary modules.

37. The method of claim 21, wherein the task module in the step of connecting further comprises a message processor that provides a messaging interface between the task manager and the task module.

38. The method of claim 21, further comprising the step of structuring a plurality of task modules in communication with a plurality of task managers according to a tree topology.

39. The method of claim 38, wherein the plurality of task managers in the step of structuring include a first task manager and a second task manager that are in direct communication with one another to facilitate completion of the task.

40. The method of claim 21, further comprising the step of providing a plurality of task modules operatively communicating over respective ports with the task manager, which task manager receives input data and manages processing of the input data into output data by selectively routing input data information in the form of unique task messages through one or more of the plurality of task modules to generate the output data.